Computer Standards & Interfaces 48 (2016) 112–123

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

An Ontology for ISO software engineering standards: 2) Proof of concept and application

C. Gonzalez-Perez [a,*], B. Henderson-Sellers [b], T. McBride [c], G.C. Low [d], X. Larrucea [e]

[a] Incipit CSIC, Spain
[b] Australia
[c] University of Technology, Sydney, Australia
[d] University of New South Wales, Australia
[e] Tecnalia, Spain

ARTICLE INFO

Article history:
Received 1 December 2015
Received in revised form 27 April 2016
Accepted 28 April 2016
Available online 14 May 2016

Keywords:
Software engineering
Ontologies
SC7
ISO/IEC 24744
Proof of concept

ABSTRACT

Software engineering standards often utilize different underpinning metamodels and ontologies, which sometimes differ between standards. For better adoption by industry, harmonization of these standards by use of a domain ontology has been advocated. In this paper we apply this approach in a proof of concept project. We recommend the creation of a single underpinning abstract domain ontology, created from existing ISO/IEC standards including ISO/IEC 24744 and 24765 and supplemented by any other sources authorized by SC7 as being appropriate and useful. Such an adoption of a single ontology will permit the re-engineering of existing International Standards such as 12207, 15288 and 33061 as refinements from this domain ontology so that these variously focussed standards can all inter-operate.

1. Introduction

The need for ISO International Standards to be consistent with each other in terms of terminology, structure and semantics has long been recognized and debated. The research reported here offers a potential harmonization across ISO software engineering standards (within Sub-Committee 7 (SC7) of ISO/IEC within Joint Technical Committee 1 (JTC1)) by creating a core set of concepts and their relationships, effectively creating an abstract domain ontology for software engineering standards creation and utilization. A domain ontology is often represented as a model, which is typically represented by a UML class diagram in which the classes are concepts in the model [1]. Indeed, the use of ontologies in software engineering has been growing over the last decade (e.g. [2–6]) although Smith [6] notes that there is a distinction, often overlooked, between philosophical ontology (its roots) and ontologies as used in information science. At the same time, the use of ontologies in standards and standardization has also been shown to be useful by e.g. [7,8].

Although Rout [9] analysed a number of SC7 standards, documenting in which standards identified terms occurred together with their (disparate) definitions, only recently has harmonization become a crucial action point (rather than a vague concern) in SC7, especially under Special Working Group 5 (SWG5).

Harmonizing concepts and maintaining consistent terminology within the current corpus of standards may be manageable, albeit not easily, when SC7 projects are tightly related. However, harmonization becomes increasingly difficult as the number and disparity of projects increases, introducing possible clashes of concepts across projects — either locally or inter-project. It can also be argued that the current SC7 corpus does not cover all topics (hence not all concepts) in the domain of software engineering — yet without an agreed base of software concepts, relations and core principles/abstractions, extension of the corpus will lead to further fragmentation and inconsistency.

Consequent to the report of McBride et al. [10], a study group was created under SWG5 of ISO/IEC's joint Sub-Committee 7 (SC7), which is responsible for all ISO software engineering standards. An infrastructure was recommended and its research aspects discussed by Henderson-Sellers et al. [11]. Building on this earlier paper, which provided the overall approach, we now provide a proof-of-concept development to complement the proposals in [11] by developing it into a plausible DEO. The benefits of conducting a proof-of-concept exercise are threefold. First, it fulfils the requirements stipulated by the SWG5, which explicitly required a proof-of-concept to be developed. Second, it shows what a DEO would look like to parties not used to working with ontologies or domain models. Third, it identifies problems and potential barriers to the development of a real DEO. By identifying problems and barriers, a future New Work Item for the creation of an SC7 ontology would be able to consider them upfront.

* Corresponding author at: Incipit CSIC. Avda. de Vigo, s/n. 15705 Santiago de Compostela. Spain.

http://dx.doi.org/10.1016/j.csi.2016.04.007
0920-5489/

Over recent years, ISO/IEC 24744[1] (Software Development Metamodel for Development Methodology) (ISO/IEC, 2007) was created specifically to formalize and standardize a set of definitions of method-focussed terms and their interrelationships, effectively creating a domain ontology (e.g. [12]). At the same time, SWG5 focussed on understanding the potential roles of not only ISO/IEC 24744 but also the Software Engineering Vocabulary (SEVOCAB: ISO/IEC 24765), which merely collects all disparate uses of every software engineering term in SC7, but makes no attempt to harmonize them. Nevertheless, SEVOCAB constitutes a useful and comprehensive data source input to the SWG5 harmonization project to create a domain ontology for software engineering standards [11]. Similarly, ISO/IEC TR 24774 presents guidelines for process descriptions that also constitute a useful source for the harmonization effort.

This paper is structured as follows. In Section 2, we briefly summarize the overall approach detailed in our earlier paper. Section 3 presents those elements identified as likely content for the proposed domain ontology — mostly derived from the two aforementioned standards (ISO/IEC 24744 and 24,765). Section 4 explains how the domain ontology and other related artefacts are intended to be used. Section 5 delivers further discussion on the practicalities of this approach, and Section 6 summarizes the project results to date.

2. Brief overview of proposed approach for the domain ontology for software engineering standards

2.1. Definitional elements ontology (DEO)

Our ontological work starts from the premise that a core set of concepts that are relevant in software engineering can be identified and characterized — for example, Process, Software Item or Team. With careful definition and the identification of inter-concept relationships, this core set forms the basis for an abstract domain ontology that does not provide specific details about any one International Standard. This is the *definitional elements ontology* (DEO), the most infrastructural piece that was described in detail in our earlier paper [11].

2.2. Configured definitional ontology (CDO)

A configured definitional ontology (CDO) is a refinement of the DEO specifically oriented towards a particular purpose — within the overall scope of SC7 software engineering standards. It could therefore be considered as the definition of a domain specific modelling language for that context (e.g. [13,14]). A CDO thus created may in fact become an international standard itself (essentially this is what the current ISO/IEC 24744 is for the context of software engineering methodologies) or may be part (probably as an Informative Annex) of an International Standard.

Since a CDO is constructed by refinement or tailoring of the DEO, this specificity can be obtained by either removing any unwanted elements of the DEO and/or introducing new concepts that refine those in the DEO (as illustrated in Fig. 1 and summarized below). Section 2.3 describes these mechanisms in full detail.

2.3. Refining the DEO into a CDO

2.3.1. Discarding elements from the DEO

Since the DEO must be expressed at a very high level of abstraction, it will likely include concepts that cover a wide range of subfields of software engineering. It is thus very unlikely that a particular CDO needs to consider all those concepts, so one of the proposed refinement mechanisms is that of element removal. Once a relevant class in the DEO has been identified (i.e. one that must be reused in the CDO), any associations with minimum cardinality greater than zero on the opposite end may be "severed" and thus a portion of the DEO discarded.

[1] All ISO standards cited are listed separately at the end of the paper.

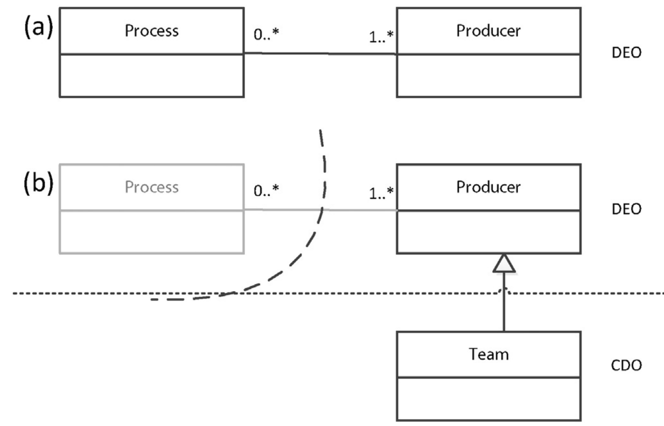

Fig. 1. Mechanisms for tailoring the DEO and obtaining CDOs from it. (a) Shows a fragment of a conceivable DEO in its original form. (b) Shows the DEO as being tailored into a particular CDO: a section of the DEO is being discarded and a subclass has been added. (after [11]).

For example, Fig. 1(a) shows a possible DEO including classes Process and Producer plus an association between them with a zero-to-many cardinality on the Process side. In a CDO targeting an area where people are relevant but processes are not, and since the DEO allows the fact that any particular producer (such as software developer Jane) may participate in no processes at all, Fig. 1(b) shows how the obtained CDO eliminates the Process element by "severing" the mentioned association, leaving only the Producer class. Contrariwise, a CDO focussing on processes but not interested in people would not be able to discard the Producer class since, according to Fig. 1(a), every process needs at least one associated producer. Having a shared domain ontology in place ensures that these constraints are observed.

2.3.2. Adding elements to the CDO

Also, and by being highly abstract, the DEO is supposed to contain only very high-level and therefore general concepts. It is thus likely that specialized definitions are needed for specific CDOs that are not present in the DEO. In cases like this, it may be necessary to create a new class for the CDO being created. However, it is important to note that any such introduction must not contradict any existing semantics in the DEO. In order to ensure that this is the case, additions must be made by specialization (is-a-kind-of) of an existing DEO concept. Fig. 1(b) shows, as an example, a specialization of Producer that has been named Team, based on the grounds that teams are producers. In the rare case that there is no existing supertype in the DEO that is suitable to serve as an abstract concept (i.e. a supertype) for the desired new one, those responsible for the maintenance of the DEO would need to consider whether such an addition – at the DEO level rather than the CDO level – is warranted. This would necessarily be a rare occurrence perhaps engendered by the inclusion of a brand-new and highly novel technology and its associated concepts.

2.4. Chaining CDOs

In the SC7 context, International Standards often fall into "families" (i.e. a collection of thematically related standards), such that the CDOs for each member of the family are similar in some way. Thus hierarchical refinement (through specialization — as above) may create a coherent set of CDOs. Fig. 2 shows a tailoring of the DEO, first into two CDOs corresponding to two working groups (WG) of SC7. Then, within each working group there may be coherent collections of standards forming intra-working group families of standards (FoS). Further refinement can then lead to CDOs for individual standards within each family. This is called here "chaining of CDOs", by which specificity and detail is added incrementally and in parallel to the organizational structure where the ontologies are going to be employed. At any point in the